AFRL-HE-WP-TR-2001-0151

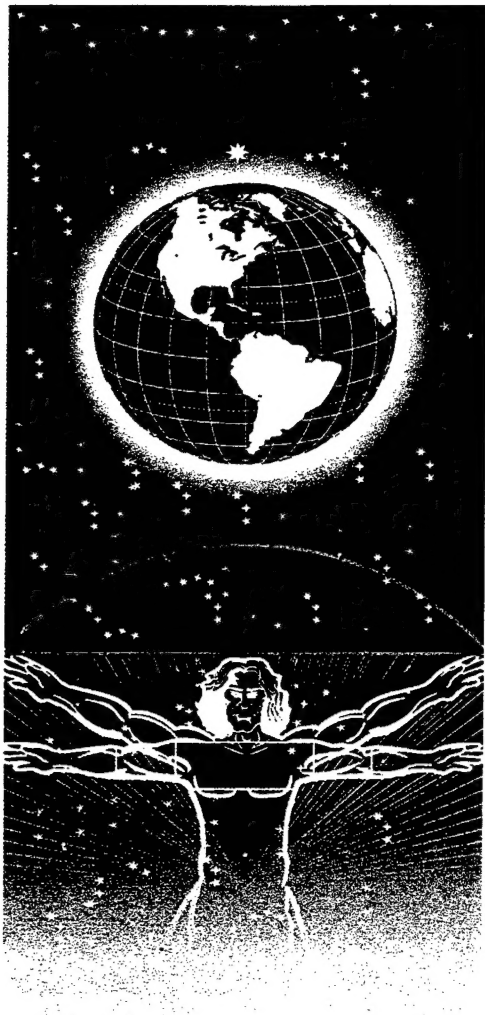# UNITED STATES AIR FORCE
RESEARCH LABORATORY

COMBAT AUTOMATION REQUIREMENTS TESTBED (CART): AN EXAMPLE APPLICATION

Thomas R. Carretta

HUMAN EFFECTIVENESS DIRECTORATE
CREW SYSTEM INTERFACE DIVISION
WRIGHT-PATTERSON AFB OH 45433-7022

Jeffrey A. Doyal
Karen A. Craig

SCIENCE APPLICATIONS INTERNATIONAL
CORPORATION
4031 COLONEL GLENN HIGHWAY
BEAVERCREEK, OH 45431

AUGUST 2001

Human Effectiveness Directorate
Crew System Interface Division
2255 H Street
Wright-Patterson AFB OH 45433-7022

20011210 018

NOTICES

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Please do not request copies of this report from the Air Force Research Laboratory. Additional copies may be purchased from:

<div style="text-align:center">

National Technical Information Service
5285 Port Royal Road
Springfield, Virginia 22161

</div>

Federal Government agencies and their contractors registered with the Defense Technical Information Center should direct requests for copies of this report to:

<div style="text-align:center">

Defense Technical Information Center
8725 John J. Kingman Road, Suite 0944
Ft. Belvoir, Virginia 22060-6218

</div>

TECHNICAL REVIEW AND APPROVAL

<div style="text-align:center">

AFRL-HE-WP-TR-2001-0151

</div>

This report has been reviewed by the Office of Public Affairs (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

MARIS M. VIKMANIS
Chief, Crew System Interface Division
Air Force Research Laboratory

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE August 2001	3. REPORT TYPE AND DATES COVERED Interim Report: March - July 2001

4. TITLE AND SUBTITLE

Combat Automation Requirements Testbed (CART): An example Application

5. FUNDING NUMBERS

PE: 62202F
TR: 7184
TA: 09
WU: 01

6. AUTHOR(S)
Thomas R. Carretta*
Jeffrey A. Doyal**
Karen A. Craig**

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Science Applications International Corporation**
4031 Colonel Glenn Highway
Beavercreek, OH 45431

8. PERFORMING ORGANIZATION

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory*
Human Effectiveness Directorate
Crew System Interface Division
Air Force Materiel Command
Wright-Patterson AFB, OH 45433-7022

10. SPONSORING/MONITORING

AFRL-HE-WP-TR-2001-0151

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*
The Combat Automation Requirements Testbed (CART) environment is an evolving software tool used to develop human performance models and integrate them with constructive simulations (CS) to address human performance modeling questions.

An informal study was conducted to determine the level of effort required by naive users to use the CART software to answer research questions, identify areas where the software and supporting materials could be improved, and evaluate the utility of CART for addressing research questions normally tackled in virtual simulations (VS).

Although model development was time consuming, much of this activity would need to be done whether researchers were using VS or CS. Regardless of method used, it would be prudent for researchers to develop detailed mission scenarios and task sequences and make educated estimates of time and workload requirements. The difference between resource requirements for VS and CS studies occurs in the activities that come next. In VS, hardware and software must be developed and subject-matter-experts testbed. In CS, the remainder of the work is mostly software development. The level of effort depends on the desired fidelity of the model and whether or not it is to "stand alone" or interact with a VS environment.

Several usability issues should be addressed before CART is widely distributed. Perhaps most important, the CART software and supporting materials are poorly documented. This is exacerbated by the non-intuitive nature of the CART user interface. Most researchers will require substantial assistance from CART software experts to fully-utilize the software. An easy to follow user's manual, pop-up menus, and Help function would go a long way toward making CART more user friendly and accessible to researchers.

14. SUBJECT TERMS
Combat Automation Requirements Testbed
CART
constructive simulation
human performance modeling
virtual simulation

15. NUMBER OF PAGES
51

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

This page intentionally left blank.

TABLE OF CONTENTS

FIGURES

TABLES

APPENDICES

PREFACE

The Combat Automation Requirements Testbed (CART) environment is an evolving software tool used to develop human performance models and to subsequently integrate them with constructive simulations to address human performance modeling questions (e.g., the utility of operator-vehicle interface [OVI] X vs. OVI Y). The Crew System Development Branch of the Air Force Research Laboratory (AFRL/HECI) initiated an in-house effort to evaluate the utility of the CART software for assisting bench-level scientists who normally perform virtual human-in-the-loop (HITL) simulations to evaluate alternate OVI designs. If CART is useful, it could assist in the design of virtual simulation studies by reducing development costs associated with sub-optimal designs.

The primary goals of the study were to (1) determine the level of effort required by naïve users to be able to use the CART software to answer research questions, (2) identify areas where the software and supporting materials could be improved, and (3) evaluate the utility of CART for addressing research questions normally tackled in virtual simulations. To this end, the CART software was used to model the effects of three alternate OVI concepts designed to enhance pilot effectiveness in performing selected airdrop mission tasks.

Although model development is time consuming, much of this activity would need to be done whether research engineers were employing virtual or constructive simulation. Regardless of which method were used, it would be prudent for researchers to develop detailed mission scenarios and task sequences and make educated estimates of time and workload requirements. The difference between resource requirements for conducting virtual and constructive simulation studies occurs in the activities that come next. In virtual simulation studies, hardware and software must be developed and subject-matter-experts tested. Clearly, this can be both time-consuming and expensive. In constructive simulation studies, the remainder of the work is mostly software development. The amount of effort depends on the desired fidelity of the model and whether or not it is to "stand alone" or interact with a virtual simulation environment.

There are several usability issues that should be addressed before CART is widely distributed. These can be grouped into three areas: Task Network Development, Model Parameterization, and Model Reports and Data Analysis. Perhaps most important, the CART software and supporting materials are poorly documented. This problem is exacerbated by the non-intuitive nature of the CART user interface. Most bench-level scientists will require substantial assistance from CART software experts to fully-utilize the software. An easy to follow user's manual, pop-up menus, and a Help function would go a long way toward making CART more user friendly and accessible to bench-level scientists.

Despite some interface problems that may cause naïve users to become discouraged, the CART modeling environment works. Some of these interface problems will be addressed when a users manual has been published. Others will require changes to the CART software. With a combination of hands-on training and expert intervention (e.g., contractor involvement), naïve users should be able to use CART to develop human performance models to answer research questions normally addressed via costly virtual simulation studies.

COMBAT AUTOMATION REQUIREMENTS TESTBED (CART):
AN EXAMPLE APPLICATION

INTRODUCTION

The complexity of crew system design issues varies substantially from study to study. Although completely new systems are sometimes developed, more often, new capabilities are proposed for an existing system in order to enhance or extend mission performance. These system enhancements often have implications for operator workload and performance. The traditional approach to assessing the impact of new systems or system enhancements on operator performance and mission effectiveness has been to conduct human-in-the-loop (HITL) virtual simulations. These can be both time consuming and expensive, requiring hardware and software development and subject-matter-experts to serve as test participants.

Constructive Simulation

Human performance models (HPMs) and constructive simulations have been proposed as an alternative to traditional HITL virtual simulations (Defense Modeling and Simulation Office, 1995; Pew & Mavor, 1998). Proponents suggest that HPMs and constructive simulations offer the advantages of reduction in the cost of test and evaluation (e.g., analysis of requirements/alternatives, cost/benefit studies, design, training) and moving test and evaluation activities earlier in the system design process, thereby reducing overall system costs. Further, supporters of constructive simulation claim that modeling and simulation is critical to the acquisition of new systems in the current environment of limited resources, shrinking budgets, and legislated reform.

It is interesting to note that the HPM and constructive simulation communities often act as both the strongest advocates and harshest critics of HPMs and constructive simulation. That is, the same people who site deficiencies in the current human performance modeling capabilities are those who advocate its use and press for continued HPM improvements. Despite the potential benefits of constructive simulation, it is often

argued that modeling software is not yet sufficiently developed to allow an authoritative representation of human behavior. As noted by McDaniel (1999), recent surveys regarding modeling and simulation problems have noted several shortcomings of HPMs. They are often narrowly focused, fail to meet user needs, are costly and time consuming to build and operate, are difficult to maintain and extend, are not easily interoperable with other HPMs, do not realistically interact with the situation or environment, and lack the ability to optimize resource interactions. Though both the Defense Modeling and Simulation Office (1995) and the National Research Council (Pew & Mavor, 1998) are strong advocates of HPMs, they site the need for better HPM environments. Given this conflicting picture of the utility of HPMs and constructive simulation, it is appropriate to ask whether sufficient gains are being made in HPM technology/application to be of use to those who would otherwise use virtual simulation.

Purpose

The Combat Automation Requirements Testbed (CART; Brett, Doyal, Malek, Martin, & Hoagland, 2000) human performance modeling environment is an evolving software tool used to develop HPMs and to subsequently integrate them with constructive simulations to address human performance modeling questions (e.g., the utility of operator-vehicle interface [OVI] X vs. OVI Y). The Crew System Development Branch of the Air Force Research Laboratory (AFRL/HECI) initiated an in-house effort to evaluate the utility of the CART software for assisting bench-level scientists who normally perform virtual HITL simulations to evaluate alternate OVI designs. If CART is useful, it could assist in the design of virtual simulation studies by reducing development costs associated with sub-optimal designs.

The primary goals of the current study were to (1) determine the level of effort required by naïve users to be able to use the CART software to answer research questions, (2) identify areas where the software and supporting materials could be improved, and (3) evaluate the utility of CART for addressing research questions normally tackled in virtual simulations (i.e., value added). To this end, the CART software was used to model the effects of three alternate OVI concepts designed to enhance the pilot's ability to perform

2

selected airdrop mission tasks (Barbato, 2000). A secondary goal was to compare results from the constructive simulation with those obtained in a previous HITL virtual simulation study (Barbato, 2000).

In the sections below, we will provide a brief overview of the CART software and its capabilities, outline the HITL virtual simulation study that provided the framework for the CART software evaluation, describe the process for creating the HPM, and finally, present an assessment of the current utility and usability of the CART software.

COMBAT AUTOMATION REQUIREMENTS TESTBED (CART)

Task Network Development

The CART human performance modeling environment is built on the US Army's proven "IMPRINT" modeling tool. It is a task network modeling environment that allows modelers to represent human behavior in terms of the tasks and functions an operator performs. In general, functions represent a higher level of decomposition and are used to combine tasks into meaningful groupings. Tasks represent the lowest level of decomposition in the model. Examples of function-level and task-level network diagrams are shown in Figures 1 and 2, respectively. Both function networks and task networks within functions are connected by a number of user-defined pathways. The modeler defines the order in which tasks and functions take place by connecting tasks and functions with a series of arrows. Often, multiple pathways can emerge out of a single function/task, representing multiple simultaneous tasks, a probabilistic decision path, or a tactical decision path that examines the state of a specified variable(s) to determine the next task/function in a sequence.
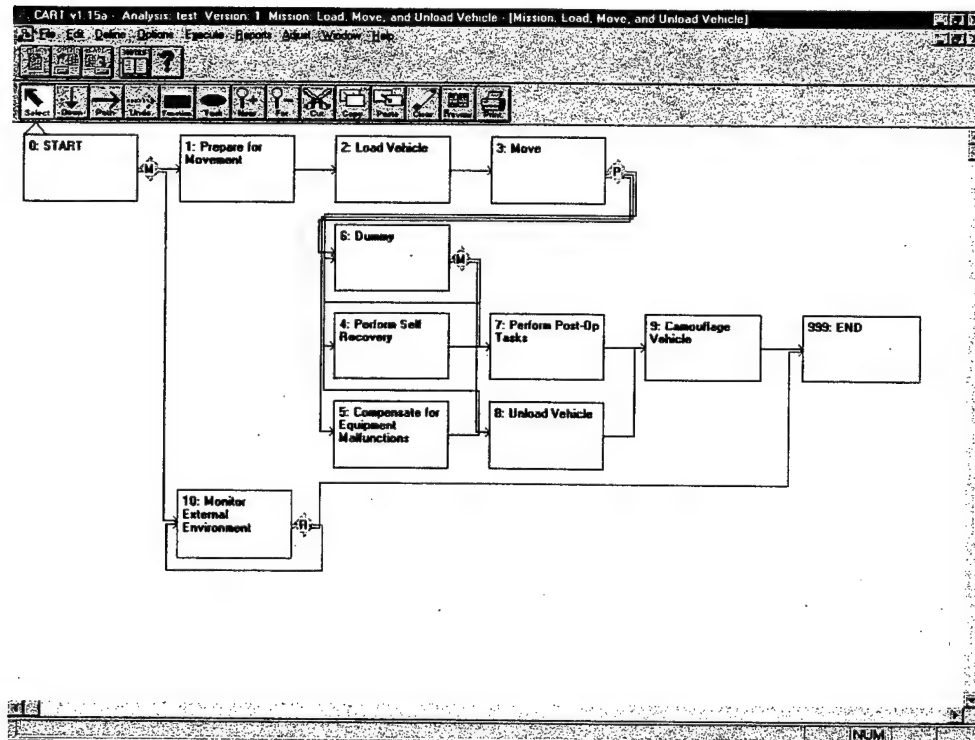
Figure 1. Example of a Function-Level Diagram in CART
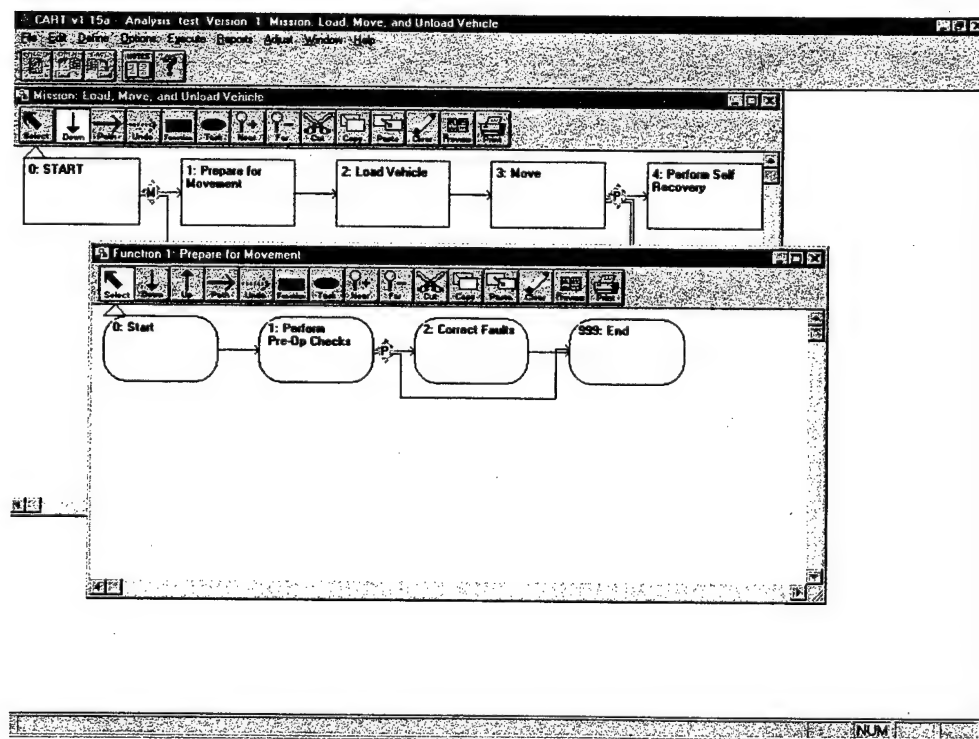
Figure 2. Example of a Task-Level Diagram in CART

Modelers can determine the content and level of detail applied in task definition as appropriate to meet their needs. For example, consider two alternate ways of describing the sequence of tasks for entering a command via a voice actuation interface. One method might be to break the sequence down into several discrete tasks (e.g., 1. Press and hold the voice actuation trigger [VAT] to activate the voice input mechanism, 2. Vocalize the message, 3. Release the VAT to signal the end of the voice message, 4. Listen to the computer response, 5. Judge the accuracy of the computer response, and 6. Repeat steps 1-5 if the response was not accurate). If, on the other hand, the modeler wanted a simpler representation of that task, steps 1-3 might be combined into a single task (voice command input), followed by a second task representing listening to (step 4) and judging the accuracy of (step 5) the computer response, then a final step for repeating the sequence, if necessary.

Model Parameterization

For any given task, several parameters can be defined by the modeler including operator task time and accuracy with a distribution and standard deviation; operator workload across the visual, auditory, cognitive, and psychomotor dimensions; model release conditions (special conditions that determine when a task should begin), and effects (used to indicate a change in status or condition) (see Brett et al., 2000 for additional details). To assist the modeler in parameterizing tasks, the CART software includes brief workload descriptors for each of the seven potential workload values in each workload category. These are based on a VACP theory of workload (McCracken & Aldrich, 1984) and are listed in Table 1. In addition, the software contains built-in "micro-models" (Micro Analysis & Design, 1999) that identify task times for a number of common operator procedures. These are shown in Table 2.

Table 1. Descriptions of Workload Levels by Category

Visual Activity
Value	Description
0.0	No Visual Activity
1.0	Visually Register/Detect (i.e., detect image)
3.7	Visually Discriminate (i.e., detect visual differences)
4.0	Visually Inspect/Check (i.e., static inspection)
5.0	Visually Locate/Align (i.e., selective orientation)
5.4	Visually Track/Follow (i.e., maintain orientation)
5.9	Visually Read (i.e., symbol)
7.0	Visually Scan/Search/Monitor (i.e., continuous)

Auditory Activity
Value	Description
0.0	No Auditory Activity
1.0	Detect/Register Sound
2.0	Orient to Sound (i.e., general orientation)
4.2	Orient to Sound (i.e., selective orientation)
4.3	Verify Auditory Feedback
4.9	Interpret Semantic Content (i.e., speech)
6.6	Discriminate Sound Characteristics
7.0	Interpret Sound Patterns (e.g., pulse rate)

Cognitive Activity
Value	Description
0.0	No Cognitive Activity
1.0	Automatic (i.e., simple association)
1.2	Alternative Selection
3.7	Sign/Signal Recognition
4.6	Evaluation/Judgment (i.e., consider a single aspect)
5.3	Encoding/Decoding, Recall
6.8	Evaluation/Judgment (i.e., consider several aspects)
7.0	Estimation, Calculation, Conversion

Psychomotor Activity
Value	Description
0.0	No Psychomotor Activity
1.0	Speech
2.2	Discrete Actuation (i.e., button, toggle, trigger)
2.6	Continuous Adjustive (i.e., flight or sensor control)
4.6	Manipulative
5.8	Discrete Adjustive (i.e., rotary, thumbwheel, lever)
6.5	Symbolic Production (i.e., writing)
7.0	Serial Discrete Manipulation (i.e., keyboard entries)

Table 2. Categories of Tasks for which Time Estimates can be Calculated

Cognitive/ Perceptual:	Psychomotor:
Eye Fixation Time	Cursor Movement with Trackball, Positioning Time
Eye Movement Time (target located at eye level)	Cursor Movement with Mouse
Decision Process	Cursor Movement with Step Keys
Listening Rate	Cursor Movement using Text Keys
Mental Rotation (visualization)	Hand Movement (Fitt's Law – Welford variant)
Perceptual Process	Head Movement Time (target located at head level)
Prioritization	
Reading Rate	Motor Process
Response Time (RT) Measures:	Pushbutton or Toggle Switch
Choice RT	Rotary Dial
Simple RT: On or Off Response	Single Finger Keying Rate
Simple RT: Physical Match	Speech
Simple RT: Name Match	Typing Rate
Simple RT: Category Match	Walking Rate
Search Time	
Terrain Association (in map reading)	

Model Reports and Data Analysis

Once a model has been created, CART can generate several reports based on execution of the model. These reports can be exported in various formats including Excel, Word, Comma-separated values, and Tab-separated values for further analysis and reduction. Preliminary summary statistics of the model functions and the tasks are generated in the "Function Performance" and 'Task Performance' reports, respectively. In addition, workload information is reported in several formats. The workload reports, "Operator Workload," "Operator Overload," and "Task Overload" are intended to indicate areas of concern for operator overload. Figure 3 shows a screenshot of a Function Performance report.

Function Performance

July 27, 2001

System 2
Mission 74 Prepare Vehicle, Engage Targets, Perform Post-Op

Function: 1   **Perform Pre-Op Checks**
Number of times performed: 6

	Standard	Minimum	Maximum	Mean	Std. Dev.	% Met
Time <=	00:14:30	00:10:29.94	00:13:43.02	00:12:25.14	00:01:27.18	100.00

Function: 2   **Boresight Weapons**
Number of times performed: 6

	Standard	Minimum	Maximum	Mean	Std. Dev.	% Met
Time <=	00:03:12	00:02:17.64	00:02:55.68	00:02:32.94	00:00:14.82	100.00

Function: 3   **Load Weapons**
Number of times performed: 6

	Standard	Minimum	Maximum	Mean	Std. Dev.	% Met
Time <=	00:02:24	00:01:38.58	00:02:01.62	00:01:50.76	00:00:10.20	100.00

Function: 4   **Move to Fighting Position**
Number of times performed: 5

	Standard	Minimum	Maximum	Mean	Std. Dev.	% Met
Time <=	00:11:06	00:10:32.34	00:16:37.02	00:14:16.68	00:02:23.34	20.00

Function: 5   **Occupy Fighting Position**
Number of times performed: 4

	Standard	Minimum	Maximum	Mean	Std. Dev.	% Met
Time <=	00:11:00	00:03:53.82	00:08:36.30	00:06:06.00	00:01:59.82	100.00

Function: 6   **Engage Targets**
Number of times performed: 4

	Standard	Minimum	Maximum	Mean	Std. Dev.	% Met
Time <=	00:02:45	00:00:20.16	00:00:55.14	00:00:44.10	00:00:16.08	100.00

Figure 3. Screenshot of a CART Function Performance Report

CART Enhancements to IMPRINT

The task-modeling environment as described above represents a basic capability inherent in the IMPRINT tool. However, the CART program has enhanced this capability by adding two key features, a goal orientation feature and High Level Architecture (HLA) and Common Object Model (COM) interfaces. First, the goal orientation capability allows the modeler to better represent the human as an information processor. Human information processor (HIP) models are a central concept for CART. HIP models assume

that the operator adapts and organizes tasks to meet current mission demands. These demands drive the operator's goals. With the CART software, the modeler can define various operator goal states and the functions/tasks performed within those goal states. The modeler can define how and when each of the goals get triggered, the priority of a given goal relative to each of the others, and a "goal action matrix" that specifies whether a goal is suspended, aborted, or runs concurrently if another higher priority goal is triggered.

The second key capability added to the IMPRINT software under the CART program is a set of interfaces that allow the CART model to interact with other modeling environments in runtime. An HLA interface was added for the primary purpose of allowing model communication with constructive mission environments. The CART model can serve as an operator, receiving inputs about the state of the world from a constructive simulation (e.g, airspeed, altitude, range to target, missile launches, etc.); activate appropriate goal states, functions, and tasks; and then send resulting operator "commands" back to the mission environment (e.g., move throttle to afterburner, turn to a given heading, release chaff, etc.). This allows the model to interact directly with a dynamic mission environment. The HLA interface was demonstrated successfully in the CART program's Case Study 1 (Brett, Doyal, Malek, Martin, & Hoagland, 2001).

In addition, a COM interface allows the CART model to call an external HPM during runtime. For example, a task in the CART model could call on an external human performance modeling environment (e.g., ACT-R, SOAR) to calculate a particularly complex task-specific calculation. The CART model would pass current information to the external model, the external model would calculate its result, and then the result would be passed back into the CART model, which would accommodate the new information and continue its run. The COM interface, available only recently, has not yet been demonstrated in a CART Case Study. However, an effort is in progress to connect a CART Joint Strike Fighter pilot HPM with an ACT-R model.

HITL VIRTUAL SIMULATION STUDY

To evaluate the utility and usability of the CART modeling environment, a human performance model of a pilot was developed. This was a part-task model, addressing only the pilot activities associated with mission replanning. The basis for selecting this type of model was a HITL virtual simulation study conducted recently by AFRL/HECI. This study is described below.

In a virtual simulation study, Barbato (2000) examined the utility of three alternate OVI designs. These interfaces were implemented in the Transport Aircraft Cockpit (TRAC, see Figure 4) in the Crew System Integration Laboratory (CSIL) at Wright-Patterson Air Force Base, OH.
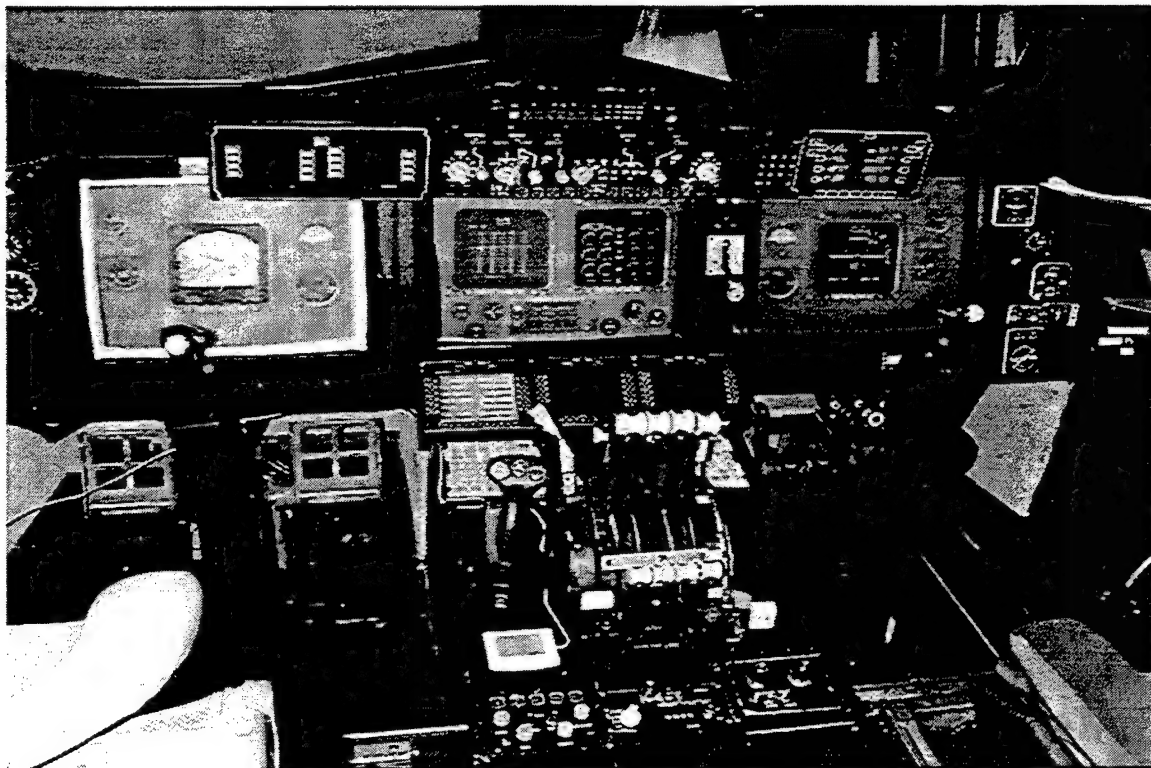

Figure 4. Transport Aircraft Cockpit (TRAC)

TRAC Facility

The TRAC is a re-configurable, three-seat (pilot, copilot, and flight engineer) transport aircraft cockpit research simulator. The out-the-window scene is displayed through wide-angle collimating windows. Head-down instrument formats are displayed using three 21-inch monitors across the front of the cockpit. Active matrix liquid crystal displays also are available as head-down display devices. Various aero- and fuel models (e.g., C-141, C-17) are available to provide realistic flying characteristics. The simulation has the flexibility to switch between a center yoke, a center C-17 style stick, or a side stick controller for flight control inputs.

Study Objective

The objective of the HITL virtual simulation study was to measure the effects of alternate control concepts on crew ability to perform selected airdrop mission tasks. Participants were trained to use three alternate OVIs. In the "baseline" minimal impact condition, the cockpit configuration included a tactical situation display and a mission computer keypad (see Figure 5). The first "enhanced display" consisted of the tactical situation display and a hand controller (see Figure 6). The second "enhanced display" consisted of the tactical situation display and voice actuation with the hand controller used for a subset of tasks (e.g., moving a waypoint). The use of voice actuation minimized the need to interact with the mission computer via either the on-screen menu system or the mission computer keypad. Mission tasking during the experiment included threat assessment and avoidance and in-flight mission/reroute planning capabilities. Information displays included large-screen head-down displays enhanced with sensor-fused information and head-up displays that depict computer-generated flight control data on an artificial outside-world scene during instrument meteorological conditions.

After training with the interfaces, each pilot "flew" six simulated missions (an "easy" and a "difficult" mission for each interface). The "easy" missions were flown at medium altitude, whereas the "difficult" missions were flown at low level. Both conditions required manual flight by the pilot, as the simulator did not contain an auto pilot mode. The mission scenarios were designed to represent a typical mission and incorporated a
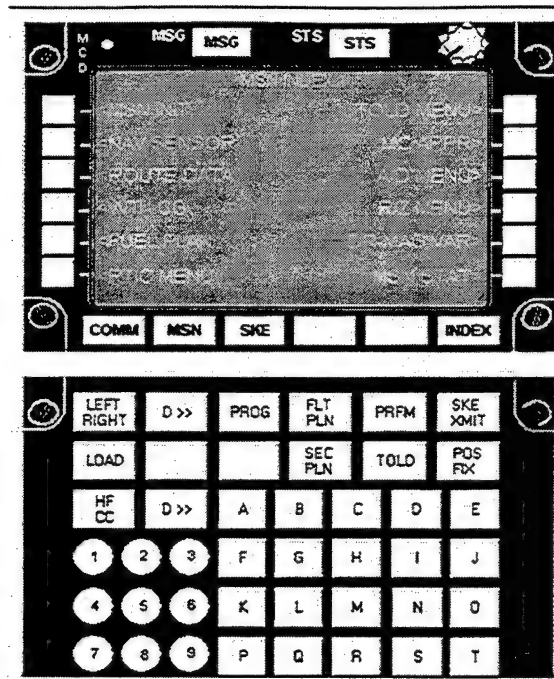
11

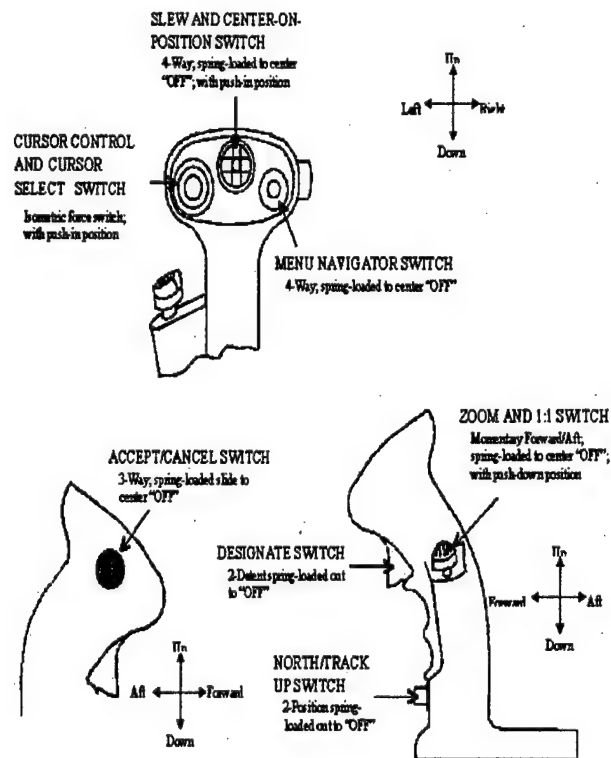**Figure 5. Mission Computer Display and Keypad**

Figure 6. Hand Controller

series of events that required the pilot to access Real-Time Information into the Cockpit (RTIC) information. Based on that information, the pilot was required to react appropriately by modifying the mission plan. The purpose of the study was to determine the extent to which these interface enhancements achieved these objectives. Objective data collected during the experiment included the time to complete mission tasks and their associated accuracies. Subjective measures of pilot situational awareness and workload also were collected.

HUMAN PERFORMANCE MODEL DEVELOPMENT

As a basis for evaluating the CART software, a CART HPM was developed that modeled mission replanning activities using the three OVIs under evaluation in the virtual simulation study. There were three main lines of activities related to this study. The first involved mission decomposition to understand the key human and system tasks and determine how these tasks were related to mission outcomes. The second line of activities was familiarization with the CART software pursuant to model development. The final line of activities consisted of building and populating the task network model and comparing the results for the three interface conditions.

Mission Decomposition

In order to develop a task sequence for each interface, it was necessary to become familiar with the TRAC cockpit and the three interface options being evaluated in the HITL virtual simulation study. This was accomplished by direct interaction with the HITL simulation in the TRAC cockpit and through discussions with engineering psychologists and software developers familiar with the HITL study protocol. An initial task sequence was developed for each interface. It then was reviewed and revised by two engineering psychologists familiar with the HITL study. The task sequence is summarized in Appendix A.

Familiarization with the CART Software

Familiarization with the CART software was achieved primarily through hands-on experience. AFRL/HECI engineering psychologists met with SAIC (Science Applications International Corporation) representatives to learn about the CART software. During this activity, several building activities were discussed and demonstrated including defining functions and tasks; entering the task sequence, decision points, and conditional probabilities; and populating the tasks with response time, response accuracy, and workload estimates. During this introduction to the CART tool, and subsequent model development, the CART interface was evaluated informally to identify potential human-computer interface issues.

Development of a CART Model

After the initial software orientation, development of the actual OVI model was begun. Using the task sequences described above, a task network model was created. This model represented pilot actions for performing five different types of mission replanning, as well as an airdrop. Further, it represented these pilot actions for the three OVIs under consideration in the HITL virtual simulation study. Finally, it represented a basic "piloting" function that accounts for the pilot scanning the flight instruments and making periodic flight control inputs, as the operators in the virtual simulation study were required to perform simultaneous flight and replanning functions.

The entire series of function and task networks included in the model is somewhat lengthy. However, we have included a single function network (Mission Replanning) and a single task network (SAM Threat Replan) as examples of what the OVI networks look like. See Figures 7 and 8, respectively. For each of the tasks, parameters such as time, accuracy, and workload (visual, auditory, cognitive, and psychomotor) were assigned. Often, the task timing estimates were calculated using the micro-model feature of the software. Where these data were not available/appropriate for the mission replanning tasks, subject matter experts within AFRL/HECI determined values. In addition, probabilities or decision logic were assigned to each multiple branch in the model.
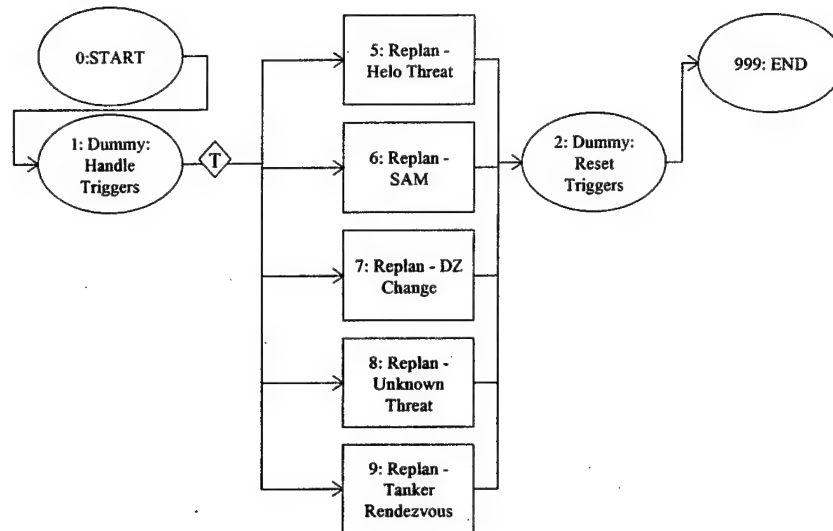
14

Figure 7. CART Diagram of a Mission Replan Sequence at the Function Level
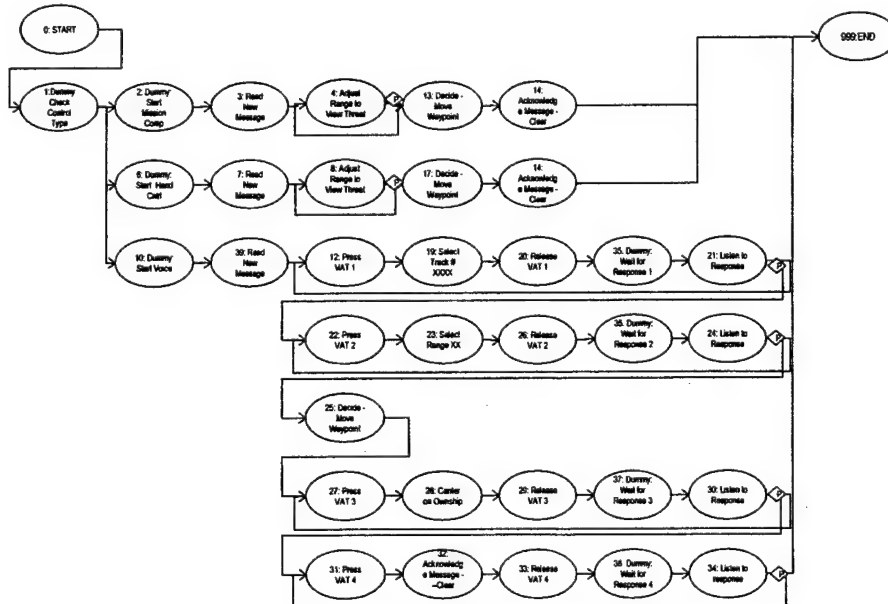

Figure 8. CART Diagram of a Mission Replan Sequence at the Task Level: Response to SAM Threat

Results

Once complete, the model was debugged and then run for a total of 150 iterations (50 iterations with each of the three OVIs). Table 3 lists the mean replanning times for three different replanning types for the constructive simulation. Although no statistical tests were conducted, the largest mean response time differences among the three interfaces in the constructive simulation occurred for the Helo Threat scenario. For the Helo Threat scenario, the Hand Controller interface required the least time (29.76 seconds), the Voice Actuation interface (35.82 seconds) was next, and the Mission Computer interface (50.94 seconds) required the greatest amount of time. It is interesting to note that the constructive simulation model suggested a different ordering of the interfaces for the Drop Zone Change and Rendezvous Change scenarios. In those scenarios the Mission Computer interface was the fastest, though the difference among the three interfaces was smaller than in the Helo Threat scenario.

Table 3. Mean Mission Replan Times for Constructive Simulation for Helo Threat, Drop Zone Change, and Tanker Rendezvous Change

Interface Type	Helicopter Threat	Drop Zone Change	Tanker Rendezvous
Mission Computer	50.94	12.30	13.02
	(10.08)	(1.68)	(3.12)
Hand Controller	29.76	20.04	18.54
	(6.72)	(7.92)	(6.24)
Voice Actuation	35.82	15.84	20.46
	(5.40)	(3.06)	(2.34)

Notes. Standard deviations are in parentheses. Results are based on 50 replications for each interface.

16

One of the goals of this study was to compare the results of the constructive simulation with those obtained in the HITL virtual simulation study (Barbato, 2000). However, we decided not to pursue this upon reviewing the virtual simulation data. Irregularities in the virtual simulation data set (e.g., outliers) and assumptions made during the virtual simulation data processing and analyses made direct comparisons with the constructive simulation difficult.

CART SOFTWARE USABILITY AND UTILITY

It is important to note that the CART software is still evolving and is not yet ready for public release. Both SAIC and Micro Analysis and Design (who are developing the CART software under subcontract to SAIC) are aware of several usability issues that have yet to be addressed. The case studies under the CART program as well as this current evaluation effort should be considered more of a "Beta test" of the software rather than a test or critique of a final product. The hope is that insights gained in this effort and the case studies will help identify any usability issues such that they can be addressed prior to final delivery.

Usability Overview

An informal evaluation of the CART interface was conducted from the perspective of a naïve user being exposed to CART for the first time. When this study was being done, the CART user's manual was not yet available. Moreover, the on-line *Help* capability had not yet been updated. As a result, much trial-and-error was needed to learn to navigate the CART interface. On a few occasions, a programmer familiar with the CART software was consulted. The evaluation below is divided into sections representing the features described in the CART Environment section of this report. The model developed during this evaluation is standalone. As a result, the HLA and external model call interfaces were not evaluated.

Task Network Development

Creating functions and tasks with the CART interface is fairly straightforward. The user first crates a new function by selecting the Function option from the toolbar, then clicking the mouse in an open space on the application window. A rectangular box representing a new function will then appear. The user can enter a function name (e.g., "Fly the Plane") by clicking on the rectangle to bring up a data entry screen. Refer back to Figure 7 for an illustration of a CART function-level diagram.

Once this is done, the user returns to the main screen in order to define a task sequence that will be performed under the function. Figure 8 shows an example of a CART task-level diagram for a surface-to-air missile threat replan sequence. The user first selects the function, and then using the toolbar, goes down one level (to the task level). The toolbar is used to create tasks in a manner similar to that for creating functions. For each task, the user indicates its workload and time requirements and the level of accuracy with which it is performed. Further, the user specifies the task network (paths and sequence of events). In some instances, *release conditions* (special conditions that determine when a task should begin) and *effects* (used to indicate a change in status or condition) need to be set in the "Effects" tab of the task. Directional arrows (→) connecting the tasks indicate the task sequence. Having more than one arrow come out of a task indicates a conditional decision. An example where this is useful is in depicting someone adjusting the screen view (zooming-in or out). After zooming-in once, the pilot would check to see if the view were satisfactory. If it were, the pilot would go on to the next task in the sequence. If it were not, the pilot would perform another zooming task. Once the task sequence has been specified, the user must enter workload, response time, and response accuracy data for each task. This is done by opening a task using the mouse, then selecting the appropriate tab (workload, response time and accuracy) and following the directions from there. Table 4 lists some of the usability issues identified during the model building process. Suggestions to correct the problems are presented for some of these items.

18

Table 4. Usability Issues in Task Network Development

Usability Issue	Description	Suggestion
Opening a file	When opening a model file, it is not apparent from the screen that the file has been opened (there is no obvious change).	Provide the user feedback that the file has been opened.
File Navigation	When opening an existing file it would be helpful if a pop-up menu similar to that found in common office software applications were available to help locate the file, change its name, and change views (e.g., function & task diagram vs. data input/ editing windows).	
Task Network Display	When a task sequence is illustrated, it is common for the lines that connect tasks to intersect. This makes it difficult to trace the task sequence.	Display separation between lines.
Goal Representation	It is difficult to represent the simultaneous occurrence of high-level goals (e.g., flying the airplane while performing a voice-actuated replan), while at the same time specifying more detailed tasks within those goal states that cannot happen at the same time (e.g., look at the HUD while looking at the TSD).	
Task Network Display	For models with more than just a few tasks, it is difficult for the user to keep track of the task sequence structure. This occurs because in long task sequences the entire sequence cannot be displayed simultaneously. The CART software has the capability to "zoom-in" or "zoom-out" to adjust the view. In very long task sequences, the font size may become too small to read after zooming-out to get the entire sequence on one screen.	

Model Parameterization

Once the task sequence has been specified, the user can enter workload, response time, and response accuracy data for each task as described earlier. This is done by opening a task using the mouse, then selecting the appropriate tab (workload, response time and accuracy) and filling in the available fields the user wishes to populate.

Table 5. Usability Issues in Model Parameterization

Usability Issue	Description	Suggestion
Response Time Calculator	The CART response time calculator does not allow the user to calculate response time for multiple steps in a task and add them together to get a total. The user must do this outside the computer interface, then manually input the data into the model.	Reprogram the calculator to allow multi-step calculations and direct data input to the model.
Default Accuracy	The default estimates for task accuracy are set to 0%.	This should be changed to 100%, as this value will be more typical in actual models.
Workload Scale	When entering workload estimates in the Goal Orientation mode, the user is limited to McCracken And Aldrich's VACP workload representation. The user cannot implement other workload scales Such as SWAT or NASA TLX. Also, the CART Interface does not allow the user to directly define The overall workload for a function or group of Tasks. Often, this is the type of data we get from HITL studies.	

The CART program allows modelers to indicate probabilities for successful completion of tasks (accuracy data). If the task is assigned a less than 100% accuracy level, a recovery path must be specified in the event that the task is not executed properly. The ending effect of a task deemed to be "inaccurate" never gets executed. If a recovery path

is not specified, the model will continue to the next task. It is very difficult to determine what type of mistake occurred (omission or commission), when the error could be discovered, and what the recovery procedure is, and then reflect this in the model. It may be more appropriate to consider this a difficult modeling problem, rather than a usability issue specific to CART. It is mentioned here because this type of problem would be obvious if it occurred in a virtual simulation study, but would be difficult to model in a constructive simulation. Table 5 lists additional concerns discovered during model parameterization.

Model Reports and Data Analysis

Generating reports in the CART modeling environment is straightforward. The user needs only to select the type of reports from a list and click the 'reports' button. Further, it is easy to export the data in several widely used formats. Currently, there is a software bug that prevents the user from setting workload thresholds; this seriously limits the utility of the workload reports. Table 6 contains a list of usability issues with the reports generated by CART.

Table 6. Usability Issues in Model Reports and Data Analysis

Usability Issue	Description	Suggestion
Total Workload	There is no default set for totaling workload in the reports. This must be set in "Options" → "Overall Workload," otherwise the total workload will be reported as zero. It is unlikely that users will want to calculate total workload by any other means than by adding the values across the four types.	This process should be set permanently to add, or at least be the default.
Workload Sampling	It would be helpful to be able to change the workload sampling scheme from event to time-based in order to facilitate the creation of graphic comparisons between interfaces.	

Usability Assessment

Despite some interface problems that may cause naïve users to become discouraged, the CART modeling environment works. Some of these interface problems will be addressed when a users manual has been published. Others will require changes to the CART software. With a combination of hands-on training and expert intervention (e.g., contractor involvement), naïve users should be able to use CART to develop human performance models to answer research questions normally addressed via costly virtual simulation studies.

The CART software is not robust. Despite meeting suggested system requirements (e.g., operating system, RAM), we were not able to run our CART model on a 200 MHz computer with 64MB RAM, a 2GB hard drive, and a Microsoft Windows 95 operating system. Mysteriously, the CART model successfully ran on a different system with a similar configuration. The CART software seems to have an inherent problem that lets it run properly on some machines and not others[1].

In addition to the items noted above, as a result of earlier CART efforts, SAIC personnel compiled a list of comments for improving the software and user interface (J. Doyal, personal communication, 7 May 2001). These comments are summarized in Table B-1. Many of the comments are related to modifications that could help users debug a CART model (e.g., presentation of a report for models that did not run properly, variable naming conventions). Other comments are concerned with CART hardware, software, and memory requirements, data input and editing (e.g., add functionality to enter data arrays, clear fields more easily, set default values for accuracy fields to 100%, instead of the current default of 0%), and inconsistencies in the software (e.g., the micro-model for calculating response time uses seconds as the unit of measurement, whereas the CART higher-order model uses minutes).

---

[1] In the period since we built and tested our CART model, Micro Analysis And Design has modified its recommendations regarding minimum RAM requirements. They now recommend at least 132 MB RAM, whereas the previous recommendation was 128 MB RAM (J. Doyal, personal communication, 27 July 2001).

Utility Assessment

Despite some interface problems that may cause naïve users to become discouraged, the CART modeling environment works. Some of these interface problems will be addressed when a users manual has been published. Others will require changes to the CART software. With a combination of hands-on training and expert intervention (e.g., contractor involvement), naïve users should be able to use CART to develop human performance models to answer research questions normally addressed via costly virtual simulation studies.

It should also be noted that the CART environment addresses some of the HPM and constructive simulation deficiencies raised by critics (e.g., Defense Modeling and Simulation Office, 1995; McDaniel, 1999; Pew & Mavor, 1998). For example, CART has both HLA and COM interfaces to allow easier integration with mission environments and integration with other HPMs. CART also has a goal-orientation capability that makes it easier to model optimized resource interactions. Moreover, because the CART software provides a relatively easy-to-use modeling *environment* (as opposed to a specific HPM), it is more likely to meet an end user's needs and budget. The goal is to provide bench-level scientists with a tool to allow them to develop an HPM specifically tailored to meet the research needs/budget of their programs.

CONCLUSION

The purpose of this study was to assess the utility of the CART software for assisting bench-level scientists who normally perform HITL virtual simulations to evaluate alternate OVI designs. To this end, we focused on determining the level of effort required by naïve users to be able to use the CART software, identifying areas where the software and supporting materials could be improved, and evaluating its utility for addressing research questions normally tackled in virtual simulations.

The level of effort needed for naïve users to learn to use the CART software is substantial. The CART software and supporting materials are poorly documented. Many bench level scientists are likely to become frustrated and either abandon using CART or require contractor intervention. Although model development is time consuming, much of this activity would need to be done whether research engineers were employing virtual or constructive simulation. Regardless of which method were used, it would be prudent for researchers to develop detailed mission scenarios and task sequences and make educated estimates of time and workload requirements. The difference in resource requirements for conducting virtual and constructive simulation studies occurs in the activities that occur next. In virtual simulation studies, hardware and software must be developed and subject-matter-experts tested. Clearly, this can be both time-consuming and expensive. In constructive simulation studies, the remainder of the work is mostly software development. The amount of effort depends on the desired fidelity of the model and whether or not it is to "stand alone" or interact with a virtual simulation environment.

Although CART software developers have addressed some of the concerns raised by the HPM and constructive simulation community, several usability issues remain. Three areas were identified for improving the CART software and supporting materials: task network development, model parameterization, and model reports and data analysis.

The user interface was non-intuitive when developing task networks. It was difficult to determine when a file had been opened and file navigation required several steps. These problems can be fixed easily by having the model appear when the file is opened and the addition of pop-up menus to assist in file navigation. Another task network interface problem is the difficulty in interpreting the task network diagrams. Overlapping lines in the diagrams make them difficult to interpret. Also, large task sequences cannot be viewed in whole without "zooming-out" which may make the image unreadable. The problem of overlapping lines can be fixed simply by separating the lines. The display of long task sequences is more problematic. It may be necessary to use large computer screens or perhaps project the task network sequences onto a large screen. A final task network issue is that the CART software does not easily emulate the simultaneous

occurrence of high-level goals when they have conflicting task-level requirements. This is not an issue in virtual simulation.

There are several instances where CART model parameterization can be improved. The response time calculator should be modified so that once a value has been calculated it can be entered directly into the model. Default estimates for task accuracy currently set to 0%, should be reset to 100%, which is more typical for most tasks. The CART software should be modified to allow use of workload models other than the VACP model and should allow workload to be defined directly for a function or group of tasks.

Model reports and data analysis could be improved by adding a default for totaling workload in the workload reports. Also, the sampling method should be changed from event-based to time-based to facilitate the graphic display and interpretation of workload.

Constructive simulation tools like CART have come a long way toward becoming valuable mechanisms for modeling human performance. Despite some interface problems that may cause naïve user to become discouraged, the CART environment works. Some problems will be addressed with the publication of a User's Guide[2], while others will require software changes.

---

[2] A draft CART User's Guide was issued after completion of this effort.

REFERENCES

Barbato, G. J. (2000). *Real-Time Information to the Cockpit: Integration for Transport Aircraft.* Wright-Patterson AFB, OH: Human Effectiveness Directorate, Crew System Interface Division.

Brett, B. E., Doyal, J. A., Malek, D. A., Martin, E. A., & Hoagland, D. G. (2000). *The Combat Automation Requirements Testbed (CART) task 1 final report: Implementation concepts and an example* (AFRL-HE-WP-TR-2000-0009. Wright-Patterson AFB, OH: Human Effectiveness Directorate, Crew System Interface Division.

Brett, B. E., Doyal, J. A., Malek, D. A., Martin, E. A., & Hoagland, D. G. (in press*). The Combat Automation Requirements Testbed (CART) case study 1 report.* Wright-Patterson AFB, OH: Human Effectiveness Directorate, Crew System Interface Division.

Defense Modeling and Simulation Office (1995). *Modeling and simulation (M&S) master plan.* Department of Defense Report 5000.59-P. Washington, DC.

McCracken, J. H., & Aldrich, T. B. (1984). *Analyses of selected LHX mission functions: Implications for operator workload and system automation goals.* US Army Research Institute Technical Report.

McDaniel, J.W. (1999). The need for better human performance models. *Proceeding of the Digital Human Modeling for Design and Engineering International Conference and Exposition.* The Hague, The Netherlands: SAE International CD-ROM.

Micro Analysis and Design (1999). Micro Saint - Human Operator Simulator (MS HOS) Version 1.8 for Unix Operating Systems - User's Manual, Chapter 12 - Micro Models.

Pew, R. W., & Mavor (Eds.) (1998). *Modeling human and organizational behavior.*
National Research Council: Washington, DC. National Academy Press.

Appendix A. HITL Simulation Task Breakdown for Each Interface Condition

---

Mission Computer (Route #1)

1. Pop-up Message Occurs: New Threat: 3 hostile helicopters appear on radar
   - Read new message
   - Range out to view threat: enter range (This may require "ranging out" more than once, if the proper range is not selected on the 1st try). **NOTE:** Range in/out can be done either by selecting a range (number) on the keypad or by using the in/out arrow keys.
   - Decide whether it is necessary to move waypoint #4 (it is necessary in this scenario)
   - Select waypoint #4 ("line select" key)
   - Select "Define/ Review Waypoint" option
   - Enter new longitude & latitude for waypoint #4
     - e.g., NxxxxxxWxxxxxxx
     - Select LL line
   - Judge accuracy of new longitude & latitude (if not acceptable, repeat previous function until acceptable)
   - Return to flight plan page (optional)
   - Acknowledge message & clear it (press message button)

2. Pop-up Message Occurs: New Threat: surface-to-air missiles detected
   - Read new message
   - Range out to view threat: enter range
   - No impact on route; so no additional action is required
   - Acknowledge message & clear it (press message button)

3. Pop-up Message Occurs: Change airdrop #1 to airdrop #2
   - Clear out airdrop #1

- o Press "Clear" button (to put Clear in scratchpad)
- o Press line select button for airdrop 1
- Enter airdrop #2 on keypad
  - o Type "AD02" (to put AD02 in scratchpad)
  - o Press line select key where route "discontinuity" appears
  - o Press "Clear" button (to put Clear in scratchpad)
  - o Press line select key where route discontinuity appears to clear discontinuity
  - ⊖ If second discontinuity is still in flight plan, press "Clear" button (to put Clear in scratchpad) and press line select key where second route discontinuity appears to clear discontinuity
- Evaluate new route and adjust as needed
- Acknowledge message & clear it (press message button)

4. Pop-up Message Occurs: New Threat: surface-to-air missiles detected

- Read new message
- Adjust display to view threat (range out)
- No impact on route; so, no additional action is required
- Acknowledge message & clear it (press message button)

5. Airdrop

- Adjust display to view airdrop
  - o Range in/out as needed
  - o Repeat as needed (pilots often repeated the above procedure several or more times during the airdrop sequence)
- Adjust airspeed, altitude, & direction as needed
- Press "Green Light" switch by seat to perform airdrop when over target
- Adjust range to preferred view
- Adjust airspeed, altitude, & direction as needed

6. Pop-up Message Occurs: Location of tanker is provided

29

- Read new message
- Clear current rendezvous point
    - Press "Clear" button (to put Clear in scratchpad)
    - Press line select button for rendezvous
- Enter new rendezvous point
    - Type "RZ0X" (to put RZ0X in scratchpad)
    - Press line select key where route discontinuity appears
    - Press "Clear" button (to put Clear in scratchpad)
    - Press line select key where route discontinuity appears to clear discontinuity
    - If second discontinuity is still in flight plan, press "Clear" button (to put Clear in scratchpad) and press line select key where second route discontinuity appears to clear discontinuity
- Acknowledge message & clear it (press message button)

Hand Controller (Route #5)

1. Pop-up Message Occurs: New Threat: 3 hostile helicopters appear on radar
- Read new message
- Activate menu (bump menu-nav switch left or right)
- Highlight menu option (bump menu-nav switch up or down X times)
- Select menu option (bump menu-nav switch right)
- Select waypoint #4 (use cursor slew switch to slew/cue/highlight waypoint 4 and depress cursor slew switch to designate)
- Move waypoint #4 (use cursor slew to "drag" waypoint to new position)
- Select new waypoint location (depress cursor slew switch to "drop" waypoint at new position)
- If new route is acceptable, press accept/reject switch forward
- If new route is unacceptable, pull accept/reject switch back and repeat required functions until acceptable route is achieved
- Accept changes (press "Dimple" switch)

- Turn re-plan off
    - If menu not up, bump menu-nav switch left or right to turn on menu
    - If menu up, bump menu-nav switch down to highlight "replan"
    - Bump menu-nav switch right, which will turn off replan and simultaneously turn off menu
- Acknowledge message & clear it (pull trigger switch)

2. Pop-up message Occurs: New Threat: surface-to-air missiles detected by radar
- Read new message
- Zoom out to look at the threat (using thumb switch)
    - To get a better view of the threat, you may want to select the threat (dimple switch) and center on it (castle switch)). Later will need to deselect the threat (castle switch, then center on ownship (castle switch).
- No threat to mission; so, no additional action is required
- Acknowledge message & clear it (pull trigger switch)

3. Pop-up Message Occurs: Drop Zone Change
- Read new message
- Select current airdrop
    - Use cursor slew switch to slew/cue/highlight any point within the pre-defined airdrop
    - Depress cursor slew switch to designate
- Bring up "Modify" menu (bump menu-nav switch left or right)
    - Highlight modify (bump menu-nav switch up or down x times to highlight "modify")
    - Bump menu-nav switch right to select "modify"
- Modify AD/RZ menu appears
    - Highlight AD (bump menu-nav switch up/down x times to highlight AD)
- AD menu appears
    - Bump menu-nav switch up/down x times until correct AD # is highlighted
    - Bump menu-nav switch right to select new AD

- If new route is acceptable, press accept/reject switch forward
- If new route is unacceptable, pull accept/reject switch back and repeat required functions until acceptable route is achieved
- Turn re-plan off
    - If menu is not on, bump menu-nav switch left or right to turn on menu
    - If menu is on, bump menu-nav switch down to highlight "replan"
    - Bump menu-nav switch right, which will turn off replan and simultaneously turn off menu)
- Acknowledge message & clear it (pull trigger switch)

4. Pop-up Message Occurs: New Threat
- Read new message
- Range out to view threat
    - To get a better view of the threat, you may want to select the threat (dimple switch) and center on it (castle switch)). Later will need to deselect the threat (castle switch, then center on ownship (castle switch).
- Decide it is not a threat; so, no additional action is required
- Acknowledge message & clear it (pull trigger switch)

5. Perform Airdrop
- Adjust display to view airdrop
    - Range zoom in/out
    - Repeat as needed (pilots often repeated the above procedure several or more times during the airdrop sequence)
- Adjust airspeed, altitude, & direction as needed
- Press "Green Light" switch by seat to perform airdrop when over target
- Range out to preferred view
    - Range zoom in/out
    - Repeat as needed (pilots often repeated the above procedure several or more times during the airdrop sequence)
- Adjust airspeed, altitude, & direction as needed

6. <u>Pop-up Message Occurs: Location of tanker is provided</u>

- Read new message
- Select current rendezvous
    - o Use cursor slew switch to slew/cue/highlight any point within the pre-defined rendezvous
    - o Depress cursor slew switch to designate
- Bring up "Modify" menu (bump menu-nav switch left or right)
    - o Highlight modify (bump menu-nav switch up or down x times to highlight "modify")
    - o Bump menu-nav switch right to select "modify"
- Modify AD/RZ menu appears
    - o Highlight RZ (bump menu-nav switch up/down x times to highlight RZ)
- RZ menu appears
    - o Bump menu-nav switch up/down x times until correct RZ # is highlighted
    - o Bump menu-nav switch right to select new RZ
- If new route is acceptable, press accept/reject switch forward
- If new route is unacceptable, pull accept/reject switch back and repeat required functions until acceptable route is achieved
- Turn re-plan off
    - o If menu is not on, bump menu-nav switch left or right to turn on menu
    - o If menu is on, bump menu-nav switch down to highlight "replan"
    - o Bump menu-nav switch right, which will turn off replan and simultaneously turn off menu)
- Acknowledge message & clear it (pull trigger switch)

<u>Voice (Route #2)</u>

NOTE: After <u>every release</u> of the Voice Activation Trigger (VAT) pilot waited for verbal feedback from voice recognition system before performing next function. Voice feedback prompt could take anywhere from 1 to 3 or 4 seconds.

1. <u>Pop-up Message Occurs: New Threat - 3 hostile helicopters appear on radar</u>

- Read new message
- Press voice activation trigger (VAT)
- "Select track number xxxxx"
- Release VAT
- Press VAT
- "Select waypoint 4"
- Release VAT
- Press VAT
- "Move"
- Release VAT
- Move waypoint 4 using control stick slew cursor
- Press VAT
- "Select" (to put the new waypoint down at desired location)
- Release VAT
- Decide if new route is acceptable
- If so, Press VAT
- "Accept changes"
- If new route not acceptable, Press VAT, "Select waypoint 4" and repeat procedure
- After new route accepted, turn re-planner off
- Press VAT
- "Replan off"
- Release VAT
- Press VAT
- "Acknowledge message" to clear it
- Release VAT

2. <u>Pop-up Message Occurs: New Threat – surface-to-air missiles detected</u>

- Read new message
- Press VAT

- "Select track xxxxx"
- Release VAT
- Press VAT
- "Select range xx"; not a threat, so no additional action required
- Release VAT
- Press VAT
- "Center on ownship"
- Release VAT
- Press VAT
- "Acknowledge message" to clear it
- Release VAT

3. Pop-up-Message Occurs: Drop Zone Change Requested
- Read new message
- Press VAT
- "Select AD 0X"
- Release VAT
- Press VAT
- "Modify to AD 0X"
- Release VAT
- Press VAT
- "Accept changes"
- Release VAT
- Press VAT
- "Replan off"
- Release VAT
- Press VAT
- "Acknowledge message"
- Release VAT

4. Pop-up Message Occurs: New Threat – surface-to-air missiles

- Read new message
- Press VAT
- "Select track xxxxx", if threat is off-screen
- Release VAT
- Press VAT
- "Select range xx"
- Release VAT
- Not a threat, no additional action required
- Press VAT
- "Center on ownship"
- Release VAT
- Press VAT
- "Acknowledge message"
- Release VAT

5. Airdrop
- Adjust display to view airdrop (range in)
    o Press VAT
    o "Range xx"
    o Release VAT
    o Repeat as needed (pilots often repeated the above procedure several or more times during the airdrop sequence)
- Adjust airspeed, altitude, & direction as needed
- Press "Green Light" switch by seat to perform airdrop when over target
- Range-out to preferred range
    o Press VAT
    o "Range xx"
    o Release VAT
    o Repeat as needed (pilots often repeated the above procedure several or more times during the airdrop sequence)
- Adjust airspeed, altitude, & direction as needed

6. <u>Pop-up Message Occurs: Location of tanker is provided</u>
- Read new message
- Press VAT
- "Range xx"
- Release VAT
- Press VAT
- "Select RD 0X"
- Release VAT
- Press VAT
- "Modify to RD 0X"
- Release VAT
- Press VAT
- "Accept changes"
- Release VAT
- Press VAT
- "Replan off"
- Release VAT
- Press VAT
- "Acknowledge message"
- Release VAT
- Range-in to preferred range
    - Press VAT
    - "Range xx"
    - Release VAT
    - Repeat as needed (pilots often repeated the above procedure several or more times during the airdrop sequence)

Table B-1. Corrections/Enhancements for 2nd delivery of CHE Software Proposed by SAIC

Table B-1. Corrections/Enhancements for 2nd delivery of CHE Software Proposed by SAIC

Item Number	Description of problem or potential enhancement	Other Notes and Status
C1	Want to be able to include array variables in snapshots (with index specified).	Will be included in second phase. Current workaround is to use temporary variables.
C2	Want to be able to include array variables in snapshots (entire arrays).	More difficult to program and possibly to use. Current workaround is to use temporary variables.
C3	On the Options menu, Review Task Data option, Set Display list, want the capability to "Clear All" fields and select a few.	Would save the user some time using this interface.
C4	Would like access to user defined macros from within tasks and within the screen for defining macros similar to current access to the Variable Catalog.	Would save the user time when defining task effects and when using macros within macros.
C5	Would like for the Interrupt Strategy for tasks to default to "Resume" instead of "Restart."	User anticipates using this strategy more often.
C6	Would like the task Mean Accuracy to default to 100% instead of 0%.	Would make probability of success default to 100% instead of 0%.
C7	Add the capability of using "Entity Arrays" in the mapping tool and in the NCI code.	This will be a large effort in the second phase.
C8	Change the database used in CART from a 16-bit application to a 32-bit application. This will address current memory issues.	This could be a very large effort and will introduce some risk.
C12.	Character Limitations: No indication of when	

Item Number	Description of problem or potential enhancement	Other Notes and Status
	number of characters in an effect/macro has exceeded the limits. You can keep typing, but only a limited number of characters get saved and you don't know there is a problem until you go to run it and it fails because a semi-colon has been truncated.	
C13.	Inconsistencies in Time Representation: Micro Models assume seconds. CART clock is in minutes. Therefore, if you use an expression from the micromodel for Time/Accuracy, you must always divide by 60 to ensure that the time will in fact be in seconds. Also, watching CART's clock in the execution monitor while running via HLA is more difficult without seeing seconds. The inconsistency problem was addressed and improved under List of Known Problems number 16 (12 is also related), but there is room for improvement. Suggestion: Anywhere time is used (i.e., time/accuracy, time/accuracy expression, external events, snapshots, system clock) use a SINGLE time representation: 00:00:00.00.	
C14	Make TRUE/FALSE system variables so that user does not have to define them.	
C15.	Add full editor window for beginning ending	

Item Number	Description of problem or potential enhancement	Other Notes and Status
	effects so that the user can see more that 3 lines at a time	
C16.	Include more powerful "nested if" capability.	{this function determines the next sensor to be used for items detected in the image, input: il_sensor (global), cns_moving, output: nxtsnsr}

if cns_moving==TRUE then   if il_sensor<=K_NSAR_HIGH then   nxtsnsr:=K_TIR_WIDE,   if il_sensor==K_TIR_WIDE then

nxtsnsr:=K_TIR_NARROW,   if il_sensor==K_TIR_NARROW then   nxtsnsr:=K_TIR_2X_NAR,   if il_sensor==K_TIR_2X_NAR then   nxtsnsr:=K_TIR_2X_NAR; |

Item Number	Description of problem or potential enhancement	Other Notes and Status

		if cns_moving==FALSE then
		
		  if
		
		il_sensor==K_WSAR_LOW
		
		then
		
		    nxtsnsr:=K_WSAR_MED,
		
		  if
		
		il_sensor==K_WSAR_MED
		
		then
		
		    nxtsnsr:=K_NSAR_LOW,
		
		  if
		
		il_sensor==K_WSAR_HIGH
		
		then
		
		    nxtsnsr:=K_NSAR_MED,
		
		  if
		
		il_sensor==K_NSAR_LOW
		
		then
		
		    nxtsnsr:=K_NSAR_HIGH;
		
		if cns_moving==FALSE then
		
		  if
		
		(il_sensor==K_NSAR_MED |
		
		il_sensor==K_NSAR_HIGH)
		
		then
		
		    nxtsnsr:=K_TIR_WIDE,
		
		  if il_sensor==K_TIR_WIDE
		
		then
		
		nxtsnsr:=K_TIR_NARROW,

Item Number	Description of problem or potential enhancement	Other Notes and Status

		if il_sensor==K_TIR_NARROW then
		
		   nxtsnsr:=K_TIR_2X_NAR,
		 if il_sensor==K_TIR_2X_NAR then
		
		   nxtsnsr:=K_TIR_2X_NAR;

C17.	Add the ability to save execution monitor views such that any variables of interest do not have to be retyped on each run.	
C18	.Need to get CARTSAINT to install and run more reliably on an NT. It has been suggested that Windows 98 is a better environment and the "suggested" environment IMPRINT/CART. However, the NCI software will not run under Windows 98. Thus, we have to run under NT. This seems to be a significant inconsistency.	
C19	Solve cut and paste problem. Eliminate "0.00" from paste buffer when NCI/CARTSAINT is running (even affects Notepad).	
C21	Add the ability to SEARCH within CART. Currently, user must generate the model, search for something in CARTSAINT, then	

Item Number	Description of problem or potential enhancement	Other Notes and Status
	go back to CART to make any changes. Also, when searching in CARTSAINT, the task window in which the search item resides opens each time an occurrence is found. However, you cannot see the specific text in the beginning/ending effect unless you expand the window. You cannot expand the window unless you CANCEL the search. Thus, for every occurrence, you must start a new search, remember which tasks you have looked at already and skip them on subsequent searches. Suggestion: Allow the user to expand the windows and see the code without canceling the search.	
C22	In the network diagram, highlight functions that are currently running, even if they are waiting on a release condition. It's a little confusing to see a function box not highlighted while its curgolstatus = 1.	
C23	Increase the number of characters that can be used in an IF statement. We attempted to use a long (not complex, or nested) IF statement, but it didn't work. Once we shortened it (breaking it into two), it seemed to work fine. What are the character limitations?	
C24	Show variable names in debug window and a	

Item Number	Description of problem or potential enhancement	Other Notes and Status
	debug statement identifier. When a debug window pops up, you can see values of up to 3 variables of interest. However, there is no labeling of which debug statement is showing, nor what variables are showing. Therefore, if a user has multiple debug statements, he must recognize which one is currently showing (which is not always easy) and then remember which particular variables he requested in that statement.	
C25	Add ability to sort in Review Task Data Window. (for, consistency, I guess this should be done in the Review Function Data and Review Goal Data windows too). Add the ability to sort by Task and by Task within Function, such that identical tasks names are listed next to each other. This will allow the user to easily check for consistency among these tasks such that all data (times, accuracy, workload values, etc.) match.	
C26	Populate reports on trials that do not run to completion. It would be helpful to have access to reports/trial data even when trials do not end normally. For example, when FRED crashes and therefore does not send the halt command, we have manually halted the trial. When this happens the reports do not get	

Item Number	Description of problem or potential enhancement	Other Notes and Status
	generated. For debugging purposes, it might be handy to have access to the trial data.	
C27	Fix "Task Branching Logic" window. Once the user begins to type the conditions for tactical branching, the "Following Node" column disappears. The user cannot see the node description and type the condition for that node at the same time. Suggestion: Leave the Following Node section and the branching condition section up at the same time, and wrap text in the condition field such that long expressions can be seen all at once.	

Item Number	Description of problem or potential enhancement	Other Notes and Status
	generated. For debugging purposes, it might be handy to have access to the trial data.	
C27	Fix "Task Branching Logic" window. Once the user begins to type the conditions for tactical branching, the "Following Node" column disappears. The user cannot see the node description and type the condition for that node at the same time. Suggestion: Leave the Following Node section and the branching condition section up at the same time, and wrap text in the condition field such that long expressions can be seen all at once.	